Energy, Mines and Resources Canada
Énergie, Mines et Ressources Canada

CANMET
Canada Centre for Mineral and Energy Technology
Centre canadien de la technologie des minéraux et de l'énergie

GRAPHER PROGRAM SOFTWARE

V. Srajer, N.J. Stuart, M. Heximer, W. Veldhuis, C. Neumiller

March, 1987

This document is published ... (CANMET)

Ce document est publié ... (CANMET)

ENERGY RESEARCH PROGRAM
COAL RESEARCH LABORATORIES
DIVISION REPORT ERP/CRL 87-19 (TR)

MICROMEDIA

87-05149/

GRAPHER PROGRAM SOFTWARE

by

V. Srajer*, N.J. Stuart**, M. Heximer***, W. Veldhuis***, C. Neumiller***

ABSTRACT

In this report, the graphing program "GRAPHER" is described. The program can produce three different types of graphs: normal distribution, linear regression, and multiple lines on the same graph. The graphs are displayed on a terminal screen that has graphics capability and plotted by the connected plotter.

---

* Head, Operations Research, CANMET, CRL, Surface Mining Lab., Devon, Alb.
** Research Scientist, CANMET, CRL, Surface Mining Lab., Devon, Alberta.
*** Co-Op Students, University of Alberta, Edmonton, Alberta.

PROGRAMME "GRAPHER"

par

V. Strajet*, N.J. Stuart**, M. Heximer***, W. Velhuis***, C. Neumiller***

RÉSUMÉ

Le présent rapport décrit le programme de production de graphiques "GRAPHER". Le programme peut produire trois types différents de graphiques: distribution normale, régression linéaire et courbes multiples sur le même graphique. Les graphiques sont affichés sur un écran doté de capacités graphiques et ils sont tracés par la table traçante connexe.

*Directeur, Recherche opérationnelle, CANMET, LRC, Lab. d'exploitation en surface, Devon (Alberta).
**Chercheur, CANMET, LRC, Lab. d'exploitation en surface, Devon (Alberta).
***Étudiants inscrits à un programme de coopération, Université de l'Alberta, Edmonton (Alberta).

CONTENT

Page

ABSTRACT..... i
RESUME..... iii
INTRODUCTION..... 1
GRAPHER PROGRAM..... 1
Sample Run..... 1
Program Listing..... 6
CONCLUSION..... 22

LIST OF FIGURES

Page

1. Straight Graph Plot..... 2
2. Normal Curve Plot..... 3
3. Normal Curve Plot with Confidence Level..... 4
4. Linear Regression Graph..... 5

INTRODUCTION

The graphing program GRAPHER can produce three different types of graphs: normal distribution, linear regression, and multiple lines on the same graph. The program was designed for the use of the Surface Mining Laboratory during statistical evaluation of data collected in the field. The program may run on VT-240 terminals, the results may be plotted on the Calcomp plotter used by the Laboratory.

GRAPHER PROGRAM

The input file of the GRAPHER program must contain the # symbol in the first column followed by the name of the line. The (x,y) coordinates of the points must then follow in the data file. The user has the option to plot symbols only or to connect the points in the graph with straight or curved lines. In the case of multiple lines in the input file, the normal and linear regression curves will only use the first set of points. The user must create separate files for each set of points when dealing with these two graphs. The plotter must have the paper inserted lengthwise, with the plotter origin set over the lower right hand corner.

Sample Run.

The program is accessed in [IAN.FALL86.MISC] directory. The program is executed by typing:

RUN GRAPHER <CR>

The user will then enter input file name containing the set(s) of points. Next, the X and Y axis length, in cm, is entered. The program will prompt for the selection of the type of the graph to be produced. The desired graph is selected by:

1 ...straight graph,
2 ...normal curve, or
3 ...linear regression.

If type 1 is chosen, the following rules are observed:

Enter the title, x-axis and y-axis names of the graph. If there is only one set of points in the data file, the user is asked if he wants the symbols plotted. Then you are prompted if the points are to be connected. If "Y" is the answer, the next step will be to type 1 for straight lines between the points, or 2 for curved lines between points. If multiple lines are to be plotted, a legend is displayed with the line names opposite their corresponding symbol.

If type 2 is chosen, the user is asked to:

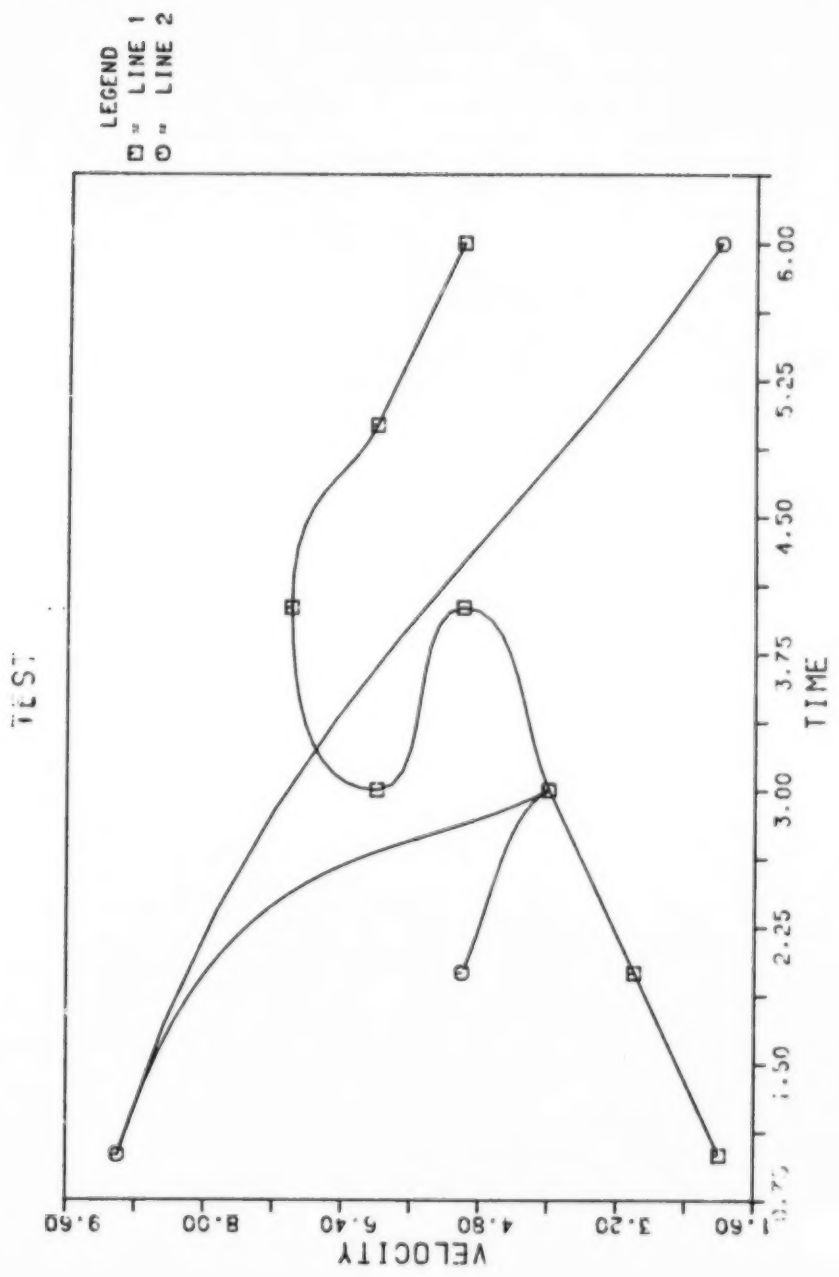

Fig. 1: Straight Graph Plot

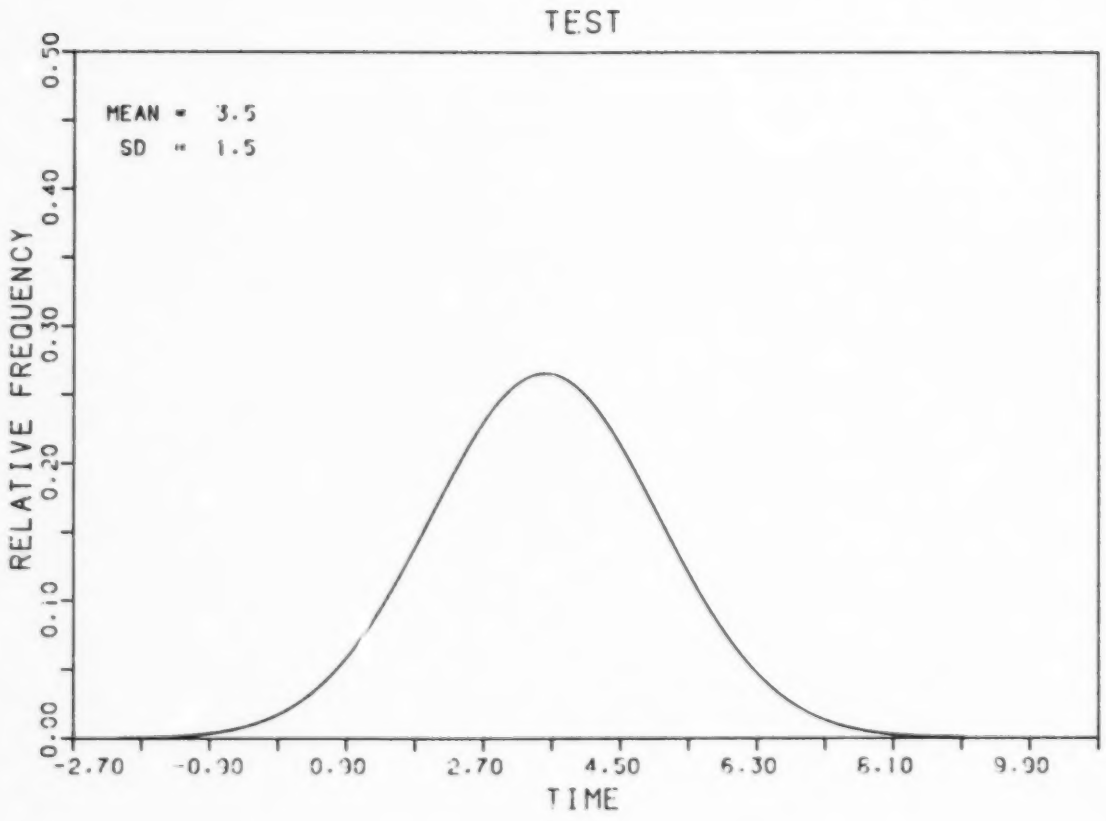

Fig. 2: Normal Curve Plot

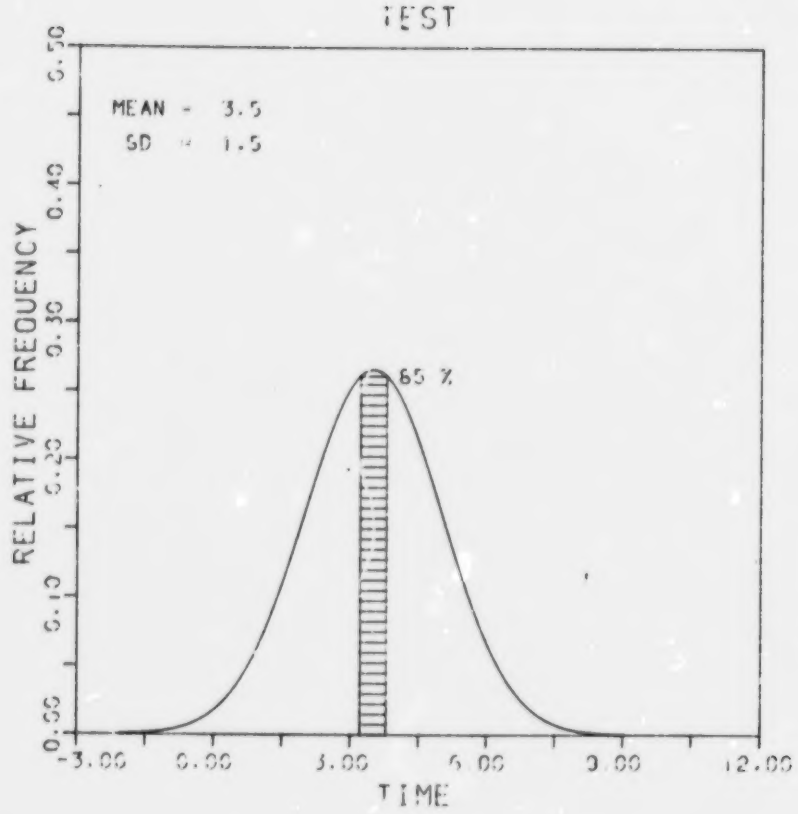

Fig. 3: Normal Curve Plot with Confidence Level

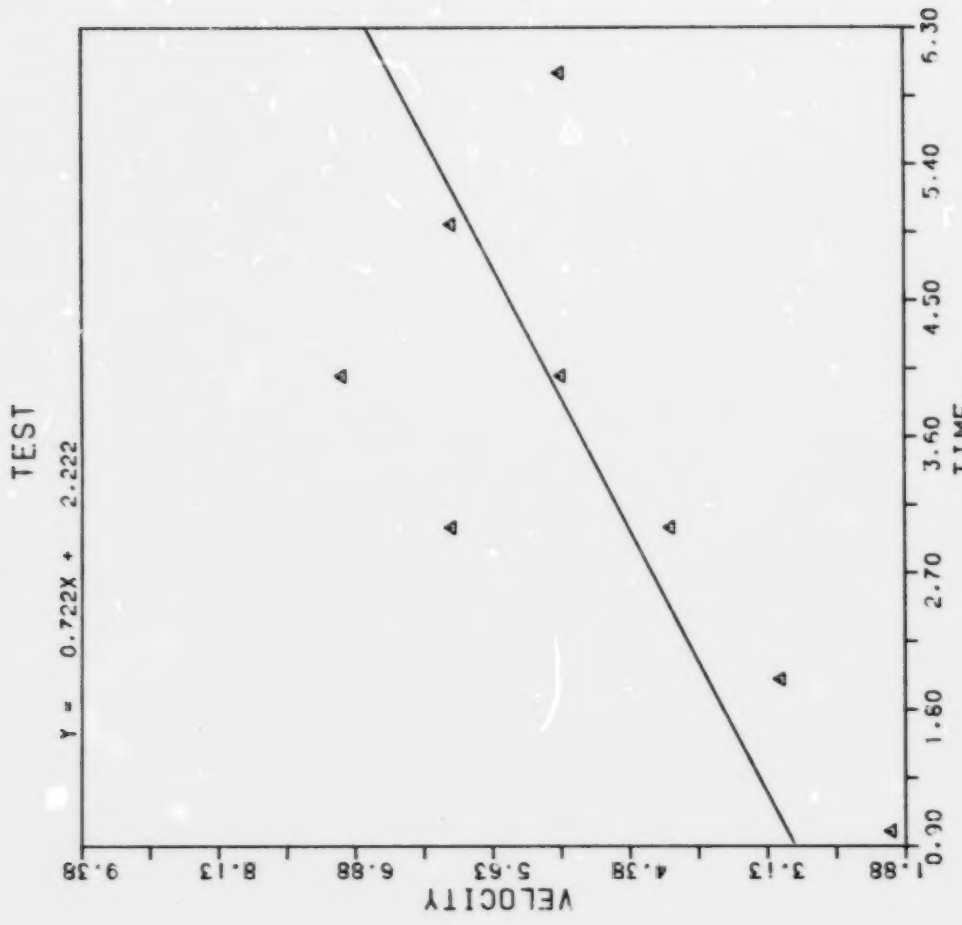

Fig. 4: Linear Regression Graph

Enter the title, x-axis name for the graph. Next, a confidence level must be specified. If 0 is entered, no confidence level is provided for the graph. The values of mean and standard deviation are then calculated and plotted on the graph.

If type 3 is chosen, the following applies:

Enter the title, x-axis and y-axis names of the graph. The program will prompt you if the symbols are to be plotted. If the answer is "Y", symbols will be plotted at the graph. The linear regression equation is fitted to the points in the format Y = AX + B, which will be plotted underneath the title of the graph.

The Figure 1 is an example of straight graph to connect plotted points by curved lines, Figures 2 and 3 are examples of normal curves, without confidence level and with confidence level of 85 % respectively, and Figure 4 is an example of linear regression graph without the equation fitted to the points.

Program Listing.

The program listing in VAX FORTRAN 77 is as follows:

```
      PROGRAM GRAPHER

      REAL X(5,500),Y(5,500),XOFF,YOFF,XLEN,YLEN,XMIN,XMAX,YMIN,YMAX
      REAL XX(500),YY(500)
      INTEGER NPTS(5),NL(5),I,J,NCHAR,NLIN
      CHARACTER*40 FNAME,LNAME(5)

      WRITE(*,*) ' INPUT DATA FILE NAME:'
      READ(*,'(A40)') FNAME
      CALL CHARCOUNT(FNAME,40,NCHAR)
      OPEN(UNIT=1,FILE=FNAME,STATUS='OLD')
      WRITE(*,*) ' ENTER THE LENGTH OF THE X-AXIS (X.XX CM):'
      READ(*,*) XLEN
      WRITE(*,*) ' ENTER THE LENGTH OF THE Y-AXIS (X.XX CM):'
      READ(*,*) YLEN
      XOFF=14.-XLEN/2.
      YOFF=11.5-YLEN/2.
      YMAX=-9.99E20
      XMAX=-9.99E20
      YMIN= 9.99E20
      XMIN= 9.99E20
      J=0
100   I=1
110   READ(1,'(A40)',END=150) FNAME
      IF(FNAME(1:1).NE.'$') THEN
              BACKSPACE(1)
              READ(1,*)X(J,I),Y(J,I)
              IF(X(J,I).GT.XMAX) XMAX=X(J,I)
              IF(X(J,I).LT.XMIN) XMIN=X(J,I)
              IF(Y(J,I).GT.YMAX) YMAX=Y(J,I)
              IF(Y(J,I).LT.YMIN) YMIN=Y(J,I)
```

```
                NPTS(J)=I
                I=I+1
                GOTO 110
        ELSE
                J=J+1
                LNAME(J)=FNAME(2:40)//' '
                CALL CHARCOUNT(LNAME(J),40,NL(J))
                GOTO 100
        ENDIF

150     NLIN=J
        WRITE(*,*) '  TYPES OF GRAPHS'
        WRITE(*,*) ' ^^^^^^^^^^^^^^^^^'
        WRITE(*,*) '[1] STRAIGHT GRAPH'
        WRITE(*,*) '[2] NORMALIZED CURVE'
        WRITE(*,*) '[3] LINEAR REGRESSION'

        WRITE(*,*) 'ENTER CHOICE:'
        READ(*,*) NUM

        IF(NUM.EQ.1) THEN
          CALL SGRAPH(X,Y,NPTS,XOFF,YOFF,XLEN,YLEN,LNAME,NL,NLIN,
        3             XMIN,XMAX,YMIN,YMAX)
        ELSE
          DO 175 J=1,NPTS(1)
            XX(J)=X(1,J)
            YY(J)=Y(1,J)
175       CONTINUE
          XX(NPTS(1)+1)=XMIN
          YY(NPTS(1)+1)=YMIN
          XX(NPTS(1)+2)=XMAX
          YY(NPTS(1)+2)=YMAX
          IF(NUM.EQ.2) CALL NCURVE(XX,YY,NPTS(1),XOFF,YOFF,XLEN,YLEN)
          IF(NUM.EQ.3) CALL LCURVE(XX,YY,NPTS(1),XOFF,YOFF,XLEN,YLEN)
        ENDIF
        END

**************

        SUBROUTINE CHARCOUNT(TEXT,LEN,NCHAR)
        INTEGER LEN,NCHAR,I
        CHARACTER TEXT*(*)

        DO 100 I=1,LEN
                IF(TEXT(I:I).NE.' ') NCHAR=I
100     CONTINUE
        RETURN
        END

***************

        SUBROUTINE SGRAPH(X,Y,NPTS,XOFF,YOFF,XLEN,YLEN,LNAME,NL,NLIN,
        3       XMIN,XMAX,YMIN,YMAX)

        REAL X(5,500),Y(5,500),XX(500),YY(500),XMIN,XMAX,YMIN,YMAX
        INTEGER NPTS(5),NL(5),SYM,NT,NX,NY,NLIN
        CHARACTER*40 TITLE,XNAME,YNAME,LNAME(5),PROMPT*1
        WRITE(*,*) ' ENTER GRAPH TITLE:'
        READ(*,10) TITLE
        CALL CHARCOUNT(TITLE,40,NT)
10      FORMAT(A40)
        WRITE(*,*) ' ENTER X-AXIS TITLE:'
        READ(*,10) XNAME
```

```
      CALL CHARCOUNT(XNAME,40,NX)
      WRITE(*,*) ' ENTER Y-AXIS TITLE:'
      READ(*,10) YNAME
      CALL CHARCOUNT(YNAME,40,NY)
      IF(NLIN.EQ.1) THEN
        WRITE(*,*) ' DO YOU WANT TO PLOT SYMBOLS (Y/N)?'
        READ(*,20) PROMPT
20      FORMAT(A1)
        SYM=0
        IF(PROMPT.EQ.'Y'.OR.PROMPT.EQ.'Y') SYM=1
      ELSE
        SYM=1
      ENDIF

      WRITE(*,*) ' DO YOU WANT TO CONNECT POINTS (Y/N)?'
      READ(*,20) PROMPT
      IF(PROMPT.EQ.'Y'.OR.PROMPT.EQ.'Y') THEN
              WRITE(*,*) ' (1) STRAIGHT LINES'
              WRITE(*,*) ' (2) CURVE'
30            WRITE(*,*) ' ENTER TYPE OF LINE (1 OR 2):'
              READ(*,*) LTYPE
              IF(LTYPE.LT.1.OR.LTYPE.GT.2) GO TO 30
      ENDIF
      XPOS=XLEN+.3
      YPOS=YLEN-1.
      CALL SCALE1(XMAX,XMIN,XLEN,XFIRST,XFACT)
      CALL SCALE1(YMAX,YMIN,YLEN,YFIRST,YFACT)
      CALL PLOTS(0,0,6)
      CALL PLOT(XOFF,YOFF,-3)
      CALL SYMBOL(XLEN/2.-(NT/2.*.3),YLEN+.5,.3,TITLE,14,0.,NT)
      CALL AXIS(0.,0.,XNAME,-NX,XLEN,0.,XFIRST,XFACT)
      CALL AXIS(0.,0.,YNAME,NY,YLEN,90.,YFIRST,YFACT)
      CALL PLOT(XLEN,0.,3)
      CALL PLOT(XLEN,YLEN,2)
      CALL PLOT(0.,YLEN,2)
      IF(NLIN.GT.1) CALL SYMBOL(XPOS+.4,YPOS+.4,.2,'LEGEND',14,0.,6)
      DO 500 J=1,NLIN
              DO 400 I=1,NPTS(J)

                      XX(I)=X(J,I)
                      YY(I)=Y(J,I)
400           CONTINUE
              XX(NPTS(J)+1)=XFIRST
              YY(NPTS(J)+1)=YFIRST
              XX(NPTS(J)+2)=XFACT
              YY(NPTS(J)+2)=YFACT
              IF(NLIN.GT.1) THEN
                CALL SYMBOL(XPOS,YPOS+.1,.2,' ',J-1,0.,-1)
                CALL SYMBOL(XPOS+.3,YPOS,.2,'=',14,0.,1)
                CALL SYMBOL(XPOS+.7,YPOS,.2,LNAME(J),14,0.,NL(J))
                YPOS=YPOS-.4
              ENDIF
              IF(SYM.EQ.1) CALL LINE(XX,YY,NPTS(J),1,-1,J-1)
              IF(LTYPE.EQ.1) THEN
                      CALL LINE(XX,YY,NPTS(J),1,0,3)
              ELSE
                      CALL CURVE(XX,YY,NPTS(J))
              ENDIF
500   CONTINUE
      CALL PLOT(0.,0.,999)

      RETURN
```

```
      END
*****************************

      SUBROUTINE LCURVE(X,Y,NPTS,XOFF,YOFF,XLEN,YLEN)

      INTEGER NPTS,I
      REAL X(500),Y(500),A,B,SUMX,SUMY,SUMXY,SUMXX,MEANX,MEANY
      CHARACTER*40 TITLE,XNAME,YNAME,PROMPT*1,FORMULA*22

      WRITE(*,*) ' ENTER GRAPH TITLE:'
      READ(*,10) TITLE
      CALL CHARCOUNT(TITLE,40,NT)
10    FORMAT(A40)
      WRITE(*,*) ' ENTER X-AXIS TITLE:'
      READ(*,10) XNAME
      CALL CHARCOUNT(XNAME,40,NX)
      WRITE(*,*) ' ENTER Y-AXIS TITLE:'
      READ(*,10) YNAME
      CALL CHARCOUNT(YNAME,40,NY)

      WRITE(*,*) ' DO YOU WANT TO PLOT SYMBOLS (Y/N)?'
      READ(*,20) PROMPT
20    FORMAT(A1)
      SYM=0
      IF(PROMPT.EQ.'Y'.OR.PROMPT.EQ.'Y') SYM=1

      SUMX=0.
      SUMY=0.
      SUMXY=0.
      SUMXX=0.
      DO 100 I=1,NPTS
        SUMX=SUMX+X(I)
        SUMY=SUMY+Y(I)
        SUMXY=SUMXY+(X(I)*Y(I))
        SUMXX=SUMXX+(X(I)*X(I))
.00   CONTINUE
      MEANX=SUMX/NPTS
      MEANY=SUMY/NPTS
      A=(SUMXY-(NPTS*MEANX*MEANY))/(SUMXX-(NPTS*MEANX**2))
      B=MEANY-A*MEANX
:     FOR LINE Y=AX+B
      CALL PLOTS(0,0,6)
      CALL PLOT(XOFF,YOFF,-3)
      WRITE(FORMULA,12) A,B
12    FORMAT('Y =',F8.3,'X +',F8.3)
      CALL SYMBOL(XLEN/2.-(NT/2.*.3),YLEN+.7,.3,TITLE,14,0.,NT)
      CALL SYMBOL(XLEN/2.-4.4,YLEN+.1,.2,FORMULA,14,0.,22)
      CALL SCALE1(X(NPTS+2),X(NPTS+1),XLEN,XFIRST,XFACT)
      CALL SCALE1(Y(NPTS+2),Y(NPTS+1),YLEN,YFIRST,YFACT)
      CALL AXIS(0.,0.,XNAME,-NX,XLEN,0.,XFIRST,XFACT)
      CALL AXIS(0.,0.,YNAME,NY,YLEN,90.,YFIRST,YFACT)
      CALL PLOT(XLEN,0.,3)
      CALL PLOT(XLEN,YLEN,2)
      CALL PLOT(0.,YLEN,2)
      X(NPTS+1)=XFIRST
      Y(NPTS+1)=YFIRST
      X(NPTS+2)=XFACT
      Y(NPTS+2)=YFACT
      IF(SYM.EQ.1) CALL LINE(X,Y,NPTS,1,-1,2)
      X1=XFIRST
      Y1=A*X1+B
      IF(Y1.LT.YFIRST) THEN
```

```
              X1=(YFIRST-B)/A
              Y1=YFIRST
            ENDIF
            X2=XFIRST+XLEN*XFACT
            Y2=A*X2+B
            YMAX=YFIRST+YLEN*YFACT
            IF(Y2.GT.YMAX) THEN
              X2=(YMAX-B)/A
              Y2=YMAX
            ENDIF
            CALL PLOT((X1-XFIRST)/XFACT,(Y1-YFIRST)/YFACT,3)
            CALL PLOT((X2-XFIRST)/XFACT,(Y2-YFIRST)/YFACT,2)
            CALL PLOT(0.,0.,999)

            RETURN
            END

*************************

            SUBROUTINE NCURVE(X,Y,NPTS,XOFF,YOFF,XLEN,YLEN)
            REAL X(500),Y(500),SD,MEAN,XSUM,SDSUM
            REAL FX(5000),XO,XAXIS(5000),XFIRST,XFACT,YFIRST,YFACT
            REAL XMIN,XMAX,T,Z(8)
            INTEGER NX,NT,NPTS,COUNT,CL
            CHARACTER NUM*12,TITLE*40,XNAME*40,CLN(8)*2
            DATA Z/0.063,0.126,.189,.253,.319,.385,.454,.524/
            DATA CLN/'95','90','85','80','75','70','65','60'/
            WRITE(*,*) 'ENTER TITLE OF GRAPH:'
            READ(*,10) TITLE
10          FORMAT(A40)
            WRITE(*,*) 'ENTER THE NAME OF THE X-AXIS:'
            READ(*,10) XNAME
12          DO 20 I=1,8
              WRITE(*,15) I,CLN(I)
15            FORMAT(' [',I1,'] = ',A2,'%')
20          CONTINUE
            WRITE(*,*) ' ENTER CONFIDENCE LEVEL CHOICE (0 FOR NONE):'
            READ(*,*) CL
            IF(CL.GT.8.OR.CL.LT.0) GOTO 12
            XSUM=0.0
            SDSUM=0.0
            MEAN=0.0
            DO 100 I=1,NPTS
            XSUM=X(I)+XSUM
100         CONTINUE
            MEAN=XSUM/FLOAT(NPTS)
            DO 500 I=1,NPTS
              SDSUM=(X(I)-MEAN)**2+SDSUM
500         CONTINUE
            SD=SQRT(SDSUM/FLOAT(NPTS))
            CALL PLOTS(0,0,6)
            CALL PLOT(XOFF,YOFF,-3)
            CALL CHARCOUNT(TITLE,40,NT)
            CALL SYMBOL(XLEN/2.-(NT/2.*.3),YLEN+.3,.3,TITLE,14,0.,NT)

            PI=3.14159265
            XO=3.716*SD
            COUNT=1
            XMIN=MEAN-1.-XO
            XMAX=MEAN+1.+XO
            CALL SCALE1(XMAX,XMIN,XLEN,XFIRST,XFACT)
            YMAX=1./SQRT(2.*PI*SD**2)
            YMIN=0.
```

```
      CALL SCALE1(YMAX,YMIN,YLEN,YFIRST,YFACT)
      CALL CHARCOUNT(XNAME,40,NX)
      CALL AXIS(0.,0.,XNAME,-NX,XLEN,0.,XFIRST,XFACT)
      CALL AXIS(0.,0.,'RELATIVE FREQUENCY',18,YLEN,90.,YFIRST,YFACT)
      CALL PLOT(XLEN,0.,3)
      CALL PLOT(XLEN,YLEN,2)
      CALL PLOT(0.,YLEN,2)
      WRITE(NUM,600) MEAN
600   FORMAT('MEAN = ',F4.1,' ')
      CALL SYMBOL(.5,YLEN-1.,.2,NUM,14,0.0,12)
      WRITE(NUM,610) SD
610   FORMAT(' SD  = ',F4.1,' ')
      CALL SYMBOL(.5,YLEN-1.5,.2,NUM,14,0.0,12)
      DO 1000 T=(MEAN-XO),(MEAN+XO),.05
        FX(COUNT)=EXP(-(T-MEAN)**2/(2.*SD**2))/SQRT(2.*PI*SD**2)
        XAXIS(COUNT)=T
        COUNT=COUNT+1
1000  CONTINUE
      COUNT=COUNT-1
      FX(COUNT+1)=YFIRST
      FX(COUNT+2)=YFACT
      XAXIS(COUNT+1)=XFIRST
      XAXIS(COUNT+2)=XFACT
      CALL LINE(XAXIS,FX,COUNT,1,0,0)
      IF(CL.NE.0) THEN
        H=0.15
        CALL CONFID(XFIRST,XFACT,YFIRST,YFACT,MEAN,SD,Z(CL),H,CLN(CL))
      ENDIF
      CALL PLOT(0.,0.,999)
      RETURN
      END

*******************************

      SUBROUTINE SCALE1(MAX,MIN,LEN,FIRST,FACT)

C     CALCULATES FACTOR IN 0.025*10**N INCREMENTS PER CM

      REAL MAX,MIN,LEN,FACT,FIRST,AXLEN
      LOGICAL FLAG
      FIRST=MIN
      FLAG=.TRUE.
      AXLEN=LEN
1     N=0
      FACT=(MAX-MIN)/(AXLEN)
      IF(FACT.EQ.0.) THEN
        FACT=.5
        FIRST=MIN-1.

        GOTO 120
      ENDIF
5     IF(FACT.LT.1.) GOTO 10
      N=N+1
      FACT=FACT/10.
      GOTO 5
10    IF(MOD(FACT,.025).NE.0) FACT=(AINT(FACT/.025)+1.)*.025
      FACT=FACT*10.**N
120   IF(MOD(MIN,FACT).NE.0)  FIRST=AINT(MIN/FACT)*FACT
      IF(FLAG.AND.(FIRST+(FACT*LEN).LE.MAX)) THEN
        AXLEN=AXLEN-.5
        GOTO 1
      ENDIF
      RETURN
      END
```

```
*****************************************

      SUBROUTINE CURVE(X,Y,NPTS)

C     SUBROUTINE TO PLOT CURVED LINE ONTO PLOTTER.

C     INITIALIZE VARIABLES.

      REAL X(500),Y(500),S(1001),Q,R,TH
      REAL TX(1000),TY(1000),NQ,NR,NS(1000)
      REAL A(1000),B(1000),C(1000),D(1000)
      REAL NX(1000),NY(1000),EX(1000),WHY(1000)
      INTEGER N,E,P,K,NPTS

      N=NPTS
      K=200

C     PROCESS DATA IN SEVERAL ROUTINES.

      CALL GRADE(X,Y,N,S)
      DO 25 J=1,N-1
            Q=X(J)
            R=Y(J)
            E=0
            P=0
            CALL TRANSFORM(P,X,Y,J,E,Q,R,N,TH,TX,TY,NQ,NR)
            CALL ANGSLOPE(TH,S(J),S(J+1),NS(J),NS(J+1))
            CALL SPLINE(TX(1),TX(2),TY(1),TY(2),NS(J),NS(J+1),
     3                  A(J),B(J),C(J),D(J))
            CALL SEGMENT(TX,TY,J,N,K)
            NX(K+2)=X(N+1)
            NX(K+3)=X(N+2)
            NY(K+2)=Y(N+1)
            NY(K+3)=Y(N+2)
            EX(1)=TX(1)
            WHY(1)=TY(1)
            DO 35 I=1,K
                  EX(I+1)=TX(1)+(I*(TX(E+2)-TX(1)))/K
                  WHY(I+1)=A(J)+B(J)*EX(I+1)+C(J)*(EX(I+1)**2)+
     3                     D(J)*(EX(I+1)**3)
35          CONTINUE
            E=K+1
            P=1
            CALL TRANSFORM(P,EX,WHY,J,E,NQ,NR,N,-TH,NX,NY,Q,R)

C     PLOT CURVE.

            CALL LINE(NX,NY,K+1,1,0,4)
25    CONTINUE


C     RETURN TO MAIN PROGRAM

      RETURN
      END

*****************************************

      SUBROUTINE TRANSFORM(P,X,Y,J,E,Q,R,N,THETA,TX,TY,NQ,NR)

C     SUBROUTINE TO TRANSFORM CO-ORDINATES WHEN AN ANGLE OF ROTATION
C     IS TO BE DETERMINED BY A SECOND POINT OR IS GIVEN
```

```
C       P=POINT TRANSFORM.LE.0 (THETA NOT GIVEN)
C       P=ROTATION TRANSFORM, P>0 (THETA GIVEN)
C       H=HYPOTHENUS
C       THETA=AXIS ROTATION, IN RADIANS
C       BETA=HYPOTHENUS ROTATION, IN RADIANS
C       GAMMA=TOTAL ROTATION, IN RADIANS
C       Q=X(J)=X OFFSET
C       NQ=X-OFFSET IN TERMS OF NEW ORIGIN
C       R=Y(J)=Y OFFSET
C       NR=Y_OFFSET IN TERMS OF NEW ORIGIN
C       E=# TIMES CALCULATION IS PERFORMED (=0 IF AXIS ROTATION IS TO BE
C               CALCULATED)
C       TX,TY=NEW CO-ORDINATES
C       N=# OF LINES IN DATA FILE

        REAL X(N),Y(N),H,THETA,BETA,GAMMA,TX(N),TY(N),Q,R,NQ,NR
        INTEGER J,P,E,N

C       CALCULATE ORIGIN AND END POINT OF LINE

        IF(P.LE.0)THEN
            TX(1)=X(J)-Q
            TY(1)=Y(J)-R
            TX(E+2)=SQRT(((X(J+1)-X(J))**2)+((Y(J+1)-Y(J))**2))
            TY(E+2)=0.0
        ENDIF

C       CALCULATE THETA

        IF(P.LE.0)THEN
            CALL THETAA(X(J+1),Q,Y(J+1),R,THETA)
        ENDIF

C       TRANSFORM POINTS OTHER THAN BEGINNING POINT OF SEGMENT

        IF(E.EQ.0)THEN
            GOTO 77

C       CALCULATION OF ANGLE BETWEEN ORIGINAL AXIS AND HYPOTENUSE (BETA)

        ELSE
            DO 20, I=1,E
                H=SQRT(((X(I)-Q)**2)+((Y(I)-R)**2))
                CALL BETAA(X(I),Q,Y(I),R,BETA)

C       DETERMINE THE TOTAL ROTATION OF THE LINE SEGMENT

                CALL GAMMAA(X(I),Q,BETA,THETA,GAMMA)

C       CALCULATE THE NEW X,Y CO-ORDINATES

                TX(I)=H*(COS(GAMMA))
                TY(I)=H*(SIN(GAMMA))

20          CONTINUE

C       CALCULATE THE NEW Q AND R FOR USE IF POINTS WILL LATER BE
C       TRANSFORMED BACK TO ORIGINAL CO-ORDINATES

C       IF Q AND R = 0, THEN THEY DO NOT NEED TO BE TRANSFORMED

77          IF((Q.EQ.0).AND.(R.EQ.0))THEN
```

```
                NQ=Q
                NR=R
                GOTO 83
            ENDIF

C       TRANSFORM Q AND R TO THE  NEW CO-ORDS

            H=SQRT(((0.0-Q)**2)+((0.0-R)**2))
            CALL  BETAA(0.0,Q,0.0,R,BETA)
            CALL GAMMAA(0.0,Q,BETA,THETA,GAMMA)
            NQ=H*(COS(GAMMA))
            NR=H*(SIN(GAMMA))
83      ENDIF

C       RETURN TO MAIN PROGRAM

        RETURN
        END

C*******************************************************************************

        SUBROUTINE THETAA(X,Q,Y,R,THETA)

C       SUBROUTINE TO CALCULATE THE ANGLE BETWEEN THE ORIGINAL AXIS AND THE
C       NEW AXIS

C       X,Y=DATA POINTS
C       Q,R=X AND Y OFFSETS, RESPECTIVELY
C       THETA= ANGLE OF AXIS ROTATION, IN RADIANS

        REAL X,Q,Y,R,THETA

        IF(((X-Q).EQ.0).AND.((Y-R).GT.0))THEN
            THETA=1.5707963
        ELSEIF(((X-Q).EQ.0).AND.((Y-R).LT.0))THEN
            THETA=-1.5707963
        ELSEIF((X-Q).LT.0)THEN
            THETA=(ATAN((Y-R)/(X-Q)))-3.1415927
        ELSE
            THETA=ATAN((Y-R)/(X-Q))
        ENDIF

C       RETURN TO MAIN PROGRAM

        RETURN
        END

C*******************************************************************************

        SUBROUTINE BETAA(X,Q,Y,R,BETA)

C       SUBROUTINE TO DETERMINE THE ANGLE BETWEEN THE HYPOTENUSE AND THE
C       X-AXIS

C       X,Y=DATA POINTS
C       Q,R=X AND Y OFFSETS RESPECTIVELY
C       BETA=ROTATION OF LINE SEGMENT FROM ORIGINAL AXIS, IN RADIANS


        REAL X,Y,Q,R,BETA

        IF(((X-Q).EQ.0).AND.((Y-R).GT.0))THEN
            BETA=1.570963
```

```
      ELSEIF(((X-Q).EQ.0).AND.((Y-R).LT.0.0))THEN
          BETA=-1.5707963
      ELSEIF(((X-Q).EQ.0).AND.((Y-R).EQ.0))THEN
          BETA=0.0
      ELSE
          BETA=ATAN((Y-R)/(X-Q))
      ENDIF

C     RETURN TO MAIN PROGRAM

      RETURN
      END

C*****************************************************************************

      SUBROUTINE GAMMAA(X,Q,BETA,THETA,GAMMA)

C     SUBROUTINE TO DETERMINE THE TOTAL ROTATION OF THE LINE SEGMENT FROM
C     THE NEW AXIS

C     X=DATA POINT
C     Q=X OFFSET
C     BETA= ANGLE OF ROTATION OF SEGMENT FROM ORIGINAL AXIS, IN RADIANS
C     THETA=ANGLE OF ROTATION OF NEW AXIS FROM ORIGINAL AXIS, IN RADIANS
C     GAMMA=ANGLE OF ROTATION OF SEGMENT FROM NEW AXIS, IN RADIANS

      REAL X,Q,BETA,THETA,GAMMA

      IF(X.LT.Q)THEN
          GAMMA=(BETA-THETA)-3.1415927
      ELSE
          GAMMA=BETA-THETA
      ENDIF

C     RETURN TO MAIN PROGRAM

      RETURN
      END

C*****************************************************************************

      SUBROUTINE ANGSLOPE(THETA,S1,S2,NS1,NS2)

C     SUBROUTINE TO CHANGE A SLOPE TO AN ANGLE, CONVERT THE ANGLE TO NEW
C     CO-ORDINATES, THEN CHANGE IT BACK TO A SLOPE

C     THETA=AXIS ROTATION, IN RADIANS
C     S1,S2=ORIGINAL SLOPES
C     NS1,NS2=TRANSFORMED SLOPES
C     ANGLE1,ANGLE2=ORIGINAL ANGLE OF ORIGINAL SLOPES, IN RADIANS
C     NEWAN1,NEWAN2=NEW ANGLE OF ORIGINAL SLOPES,IN RADIANS

      REAL THETA,S1,S2,NS1,NS2,NEWAN1,NEWAN2,ANGLE1,ANGLE2

C     CALCULATE ORIGINAL ANGLES

      ANGLE1=ATAN(S1)
      ANGLE2=ATAN(S2)

C     CALCULATE THE NEW ANGLES

      NEWAN1=ANGLE1-THETA
```

```
      NEWAN2=ANGLE2-THETA

C     CALCULATE THE NEW SLOPES

      NS1=TAN(NEWAN1)
      NS2=TAN(NEWAN2)

C     RETURN TO MAIN PROGRAM

77    RETURN
      END

*******************************

      SUBROUTINE SPLINE(X1,X2,Y1,Y2,S1,S2,A,B,C,D)

C     A SUBROUTINE TO FIT A CUBIC FUNCTION BETWEEN 2 POINTS

      REAL X1,X2,Y1,Y2,S1,S2,A,B,C,D,NUM,DENOM

C     CUBIC FUNCTION EQUATIONS

C     Y1=A+B*X1+C*(X1**2)+D*(X1**3)
C     Y2=A+B*X2+C*(X2**2)+D*(X2**3)

C     SLOPE FUNCTION EQUATIONS ARE THE DIFFERENTIAL OF THE CUBIC
C     EQUATIONS

C     S1=B+2*C*X1+3*D*(X1**2)
C     S2=B+2*C*X2+3*D*(X2**2)

C     MATHEMATICAL MANIPULATION PRODUCES THE FOLLOWING EQUATIONS
C     TO SOLVE FOR A,B,C,D

      NUM=(Y1-Y2)-(S2*(X1-X2))+(X2*(S1-S2))-(0.5*(X1+X2)*(S1-S2))

      DENOM=(0.5*(X2**3))-(0.5*(X1**3))+((3./2.)*(X1**2)*X2)-((1.5)
     3       *X1*(X2**2))

C     IF THE DENOMINATOR = 0 THEN D IS ASSUMED TO BE 0

      IF(DENOM.EQ.0)THEN
          D=0.
      ELSE
          D=NUM/DENOM
      ENDIF

C     IF X1=X2 THEN C IS 0

      IF((X1-X2).EQ.0)THEN
          C=0.
      ELSE
          C=((S1-S2)-(3*D*((X1**2)-(X2**2))))/(2*(X1-X2))
      ENDIF

      B=S1-(2*C*X1)-(3*D*(X1**2))

      A=Y1-(B*X1)-(C*(X1**2))-(D*(X1**3))

C     RETURN TO MAIN PROGRAM

      RETURN
      END
```

```
      SUBROUTINE MYSCALE(ARRAY,N,MIN,MAX,LENGTH,FIRST,FACT)

C     SUBROUTINE TO CALCULATE SCALING FACTOR OF AN ARRAY

      REAL ARRAY(N+2),FIRST,FACT,MAX,MIN,DELTA,LARGE,LENGTH
      INTEGER N

C     ARRAY=ARRAY FOR WHICH SCALING FACTORS ARE CALCULATED
C     FIRST=FIRST VALUE FOR AXIS
C     LARGE=ADJUSTED MAXIMUM ELEMENT OF ARRAY
C     FACT=SCALING FACTOR
C     MAX=MAXIMUM ARRAY ELEMENT
C     MIN=MINIMUM ARRAY ELEMENT
C     DELTA=UNADJUSTED SCALING FACTOR
C     N=# OF DATA POINTS IN ARRAY
C     LENGTH=AXIS LENGTH
C     SMALL=ADJUSTED FIRST VALUE USED IN FACT CALCULATION
      FACT=(MAX-MIN)/LENGTH
      IF(MIN.GT.100.0) THEN
        FIRST=AINT(MIN-FACT*.5)
        MAX=MAX+FACT
        FACT=AINT((MAX-FIRST)/LENGTH)
      ELSE
        FIRST=AINT(MIN)
        FACT=AINT(((MAX+5.0)-MIN)/LENGTH)*10.)/10.
      ENDIF
      IF(FACT.EQ.0.0) THEN
        IF(MIN.GT.1.0) FIRST=MIN-1.0
        FACT=AINT(4.0/LENGTH*10.)/10.
      ENDIF
      IF(MIN.EQ.0.0.AND.MAX.GT.100.0) FACT=AINT(FACT*1.05)

C     RETURN TO MAIN PROGRAM

      RETURN
      END
*****************************************************************

      SUBROUTINE GRADE(X,Y,N,S)

C     SUBROUTINE TO CALCULATE GRADES (SLOPES) AT CERTAIN POINTS BY USING
C     A SLOPE PARALLEL TO A SLOPE BETWEEN THE TWO ADJACENT POINTS

C     X,Y=DATA POINTS
C     N=# OF DATA LINES IN FILE
C     S=SLOPE

      REAL X(N),Y(N),S(N)
      INTEGER N

C     FOR THE SLOPE OF THE FIRST POINT
          IF((X(2)-X(1)).EQ.0)THEN
              S(1)=999999.999
          ELSE
              S(1)=((Y(2)-Y(1))/(X(2)-X(1)))
          ENDIF
      IF(N.EQ.2) GOTO 110
      DO 100 J=2,N-1
C         IF SLOPE IS INFINITY, ASSIGN A LARGE VALUE

          IF((X(J+1)-X(J-1)).EQ.0)THEN
              S(J)=999999.999
          ELSE
```

```
        S(J)=((Y(J+1)-Y(J-1))/(X(J+1)-X(J-1)))
    ENDIF
CONTINUE

CONTINUE
    IF SLOPE IS INFINITY, ASSIGN A LARGE VALUE

    IF((X(N-1)-X(N)).EQ.0)THEN
        S(N)=999999.999
    ELSE
        S(N)=((Y(N)-Y(N-1))/(X(N)-X(N-1)))
    ENDIF
RETURN TO MAIN PROGRAM

RETURN
END
*************************************************************************

SUBROUTINE ANGLE(X,Y,N,J,S)

SUBROUTINE TO CALCULATE THE SLOPE AT EACH POINT BY USING ANGLES

X,Y=DATA POINTS
N=# OF DATA LINES
S=SLOPE AT POINT
ALPHA=ANGLE OF SLOPE AT POINT
SIGMA=ANGLE BETWEEN SLOPE ANGLE AND HYPOTENUSE FORMED BY POINT
        AND POINT AGTER
RHO=ANGLE FORMED BETWEEN HYPOTENUSE AND X-AXIS
TAU=ANGLE FORMED BETWEEN HYPOTENUSE AND Y-AXIS
D=DISTANCE BETWEEN TWO POINTS (HYPOTENUSE)

REAL X(N),Y(N),S(N),D1,D2,ALPHA,RHO,TAU,PI,MU,CHI
INTEGER N,J

PI=3.1414927

FOR THE FIRST AND LAST SLOPES, A DIFFERENT CALCULATION IS MADE
BASED ON DELTA Y OVER DELTA X

FOR THE SLOPE AT THE FIRST POINT

IF(J.EQ.1)THEN
    IF((X(J+1)-X(J)).EQ.0)THEN
        S(J)=999999.999
    ELSE
        S(J)=(Y(J+1)-Y(J))/(X(J+1)-X(J))
    ENDIF
    GOTO 100

FOR THE SLOPE AT THE LAST POINT

ELSEIF(J.EQ.N)THEN
    IF((X(J)-X(J-1)).EQ.0)THEN
        S(J)=999999.999
    ELSE
        S(J)=(Y(J)-Y(J-1))/(X(J)-X(J-1))
    ENDIF
    GOTO 100
ENDIF

CALCULATE THE DISTANCE BETWEEN POINTS
```

```
      D1=SQRT(((Y(J)-Y(J-1))**2)+((X(J)-X(J-1))**2))
      D2=SQRT(((Y(J+1)-Y(J))**2)+((X(J+1)-X(J))**2))

C     CALCULATE THE ANGLES NEEDED TO CALCULATE THE SLOPE

      TAU=ATAN((X(J)-X(J-1))/(Y(J)-Y(J-1)))

      MU=ATAN((X(J+1)-X(J))/(Y(J+1)-Y(J)))

C     IF THE DISTANCE BETWEEN 1 AND 2, AND BETWEEN 2 AND 3 ARE
C     THE SAME, THE FOLLOWING CONDITIONS APPLY

      IF(D2.EQ.D1)THEN
          IF((ABS(TAU).EQ.ABS(MU)).AND.(X(J-1).EQ.X(J+1)))THEN
              S(J)=999999.999
          ELSEIF((ABS(TAU).EQ.ABS(MU)).AND.(Y(J-1).EQ.Y(J+1)))THEN
              S(J)=0.00000
          ELSE
              D2=D2+1
              WRITE(2,*),'D2=D1'
              GOTO 799
          ENDIF
          GOTO 100
799   ENDIF

C     RHO MUST HAVE THE SAME SIGN AS TAU FOR THE SIGN OF THE SLOPE TO
C     WORK OUT CORRECTLY, AND CHI IS ASSIGNED AN APPROPIATE VALUE (TAU-PI

      IF(TAU.LT.0)THEN
          RHO=-(PI/2.)-TAU
          CHI=-RHO
      ELSE
          RHO=(PI/2.)-TAU
          CHI=-RHO
      ENDIF

C     CALCULATION OF ALPHA (ANGLE CORRESPONDING TO THE SLOPE)
C     FOR EACH CASE TAU MUST BE CONSIDERED FOR FOUR DIFFERENT
C     ANGLES; 1) <PI/8
C             2) PI/8<TAU<PI/4
C             3) PI/4<TAU<3PI/8
C             4) TAU>3PI/8

C     CASE1:  Y1>Y2<Y3

1     IF((Y(J-1).GT.Y(J)).AND.(Y(J+1).GT.Y(J)))THEN
          ALPHA=(RHO*D2)/(ABS(D2+D1))

C     CASE2:  Y1<Y2>Y3

2     ELSEIF((Y(J-1).LT.Y(J)).AND.(Y(J+1).LT.Y(J)))THEN
          ALPHA=(RHO*D2)/(D1+D2)

C     CASE3A:  X1>X2>X3 AND Y1>Y2>Y3

31    ELSEIF(((X(J-1).GT.X(J)).AND.(X(J).GT.X(J+1))).AND.
     3     ((Y(J-1).GT.Y(J)).AND.(Y(J).GT.Y(J+1))))THEN
          IF(ABS(TAU).LT.(PI/8.))THEN
              ALPHA=(CHI*D2)/(ABS(D2-D1))
          ELSE
              ALPHA=(RHO*D2)/(D2+D1)
          ENDIF
```

```
C       CASE3B:  X1>X2>X3 AND Y1<Y2<Y3

32      ELSEIF(((X(J-1).GT.X(J)).AND.(X(J).GT.X(J+1))).AND.
3            ((Y(J-1).LT.Y(J)).AND.(Y(J).LT.Y(J+1))))THEN
            IF(ABS(TAU).LT.(PI/8.))THEN
                ALPHA=(CHI*D2)/(ABS(D2-D1))
            ELSE
                ALPHA=(RHO*D2)/(D2+D1)
            ENDIF


C       CASE4:  X1<X2>X3

4       ELSEIF((X(J-1).LT.X(J)).AND.(X(J+1).LT.X(J)))THEN
            IF(ABS(TAU).LT.(PI/8.))THEN
                ALPHA=(RHO*D2)/(ABS(D2-D1))
            ELSEIF(ABS(TAU).LT.(PI/4.))THEN
                ALPHA=(CHI*D2)/(ABS(D2-D1))
            ELSE
                ALPHA=(RHO*D2)/(ABS(D2+D1))
            ENDIF

C       CASE5:  X1>X2<X3

5       ELSEIF((X(J-1).GT.X(J)).AND.(X(J+1).GT.X(J)))THEN
            IF(ABS(TAU).LT.(PI/4.))THEN
                ALPHA=(CHI*D2)/(ABS(D2-D1))
            ELSEIF(ABS(TAU).LT.((3./8.)*PI))THEN
                ALPHA=(RHO*D2)/(ABS(D2-D1))
            ELSE
                ALPHA=(RHO*D2)/(ABS(D2+D1))
            ENDIF

C       CASE6:  X1<X2<X3 AND Y1<Y2<Y3

6       ELSEIF(((X(J-1).LT.X(J)).AND.(X(J).LT.X(J+1))).AND.
3            ((Y(J-1).LT.Y(J)).AND.(Y(J).LT.Y(J+1))))THEN
            IF(ABS(TAU).LT.(PI/8.))THEN
                ALPHA=(RHO*D2)/(ABS(D2-D1))
            ELSE
                ALPHA=(RHO*D2)/(D2+D1)
            ENDIF

C       CASE7:   X1<X2<X3 AND Y1>Y2>Y3

7       ELSEIF(((X(J-1).LT.X(J)).AND.(X(J).LT.X(J+1))).AND.
3            ((Y(J-1).GT.Y(J)).AND.(Y(J).GT.Y(J+1))))THEN
            IF(ABS(TAU).LT.(PI/8.))THEN
                ALPHA=(RHO*D2)/(ABS(D2-D1))
            ELSE
                ALPHA=(RHO*D2)/(ABS(D2+D1))
            ENDIF
        ENDIF

C       CALCULATE SLOPE

        S(J)=TAN(ALPHA)

C       RETURN TO MAIN PROGRAM

100     RETURN
        END
```

```
*******************************888

      SUBROUTINE SEGMENT(X,Y,J,N,K)

C     SUBROUTINE THAT DETERMINES THE NUMBER OF SEGMENTS THAT THE SPLINE
C     FUNCTION WILL SPLIT THE LINE, BETWEEN TWO GIVEN POINTS, INTO

C     X,Y=DATA POINTS
C     K=# OF SEGMENTS THE LINE BETWEEN TWO POINTS IS DIVIDED INTO
C     N=# OF DATA LINES IN FILE

      REAL X(N),Y(N)
      INTEGER N,J,K
C     THE NUMBER OF SEGMENTS DEPENDS ON THE LENGTH OF THE LINE

      IF((ABS(X(J+1)-X(J)).GT.25).OR.(ABS(Y(J+1)-Y(J)).GT.25))THEN
          K=50
      ELSEIF((ABS(X(J+1)-X(J)).GT.5).OR.(ABS(Y(J+1)-Y(J)).GT.5))THEN
          K=25
      ELSEIF((ABS(X(J+1)-X(J)).GT.1).OR.(ABS(Y(J+1)-Y(J)).GT.1))THEN
          K=15
      ELSE
          K=10
      ENDIF

C     RETURN TO MAIN PROGRAM

      RETURN
      END

*****************************************

      SUBROUTINE CONFID(XFIRST,XFACT,YFIRST,YFACT,MEAN,SD,Z,H,CH)

      REAL XFIRST,YFIRST,XFACT,YFACT,MEAN,SD,Z,H,ANGLE
      REAL LX,LY,RX,RY,YO
      CHARACTER*2 CH

      PI=3.14159265
      ANGLE=0.0
      LX=MEAN-Z*SD
      LY=EXP(-(LX-MEAN)**2/(2.*SD**2))/SQRT(2.*PI*SD**2)
      RX=MEAN+Z*SD
      RY=EXP(-(RX-MEAN)**2/(2.*SD**2))/SQRT(2.*PI*SD**2)
      LX=(LX-XFIRST)/XFACT
      LY=(LY-YFIRST)/YFACT
      RX=(RX-XFIRST)/XFACT
      RY=(RY-YFIRST)/YFACT
      YO=0.0
      CALL HRECT(LX,YO,LY,RX-LX,ANGLE,H)
      CALL SYMBOL(RX+.2,RY-.1,.2,CH//' %',14,0.,4)
      RETURN
      END

*****************************************

      SUBROUTINE HRECT(XO,YO,HEIGHT,WIDTH,ANGLE,H)

C     DRAWS A HATCHED RECTANGLE
C     IF H = 0 NO HATCHING IS DRAWN
C     IF H < 0 VERTICAL HATCHING
C     IF H > 0 HORIZONTAL HATCHING
```

```
      REAL DX,DY,HEIGHT,WIDTH,ANG,XO,YO,ANGLE,H,X

      DX=XO
      DY=YO
      ANGLE=0.0
      ANG=ANGLE*0.0174533
      CALL PLOT(DX,DY,3)

      DX=DX+WIDTH*COS(ANG)
      DY=DY+WIDTH*SIN(ANG)
      CALL PLOT(DX,DY,2)

      DX=DX-HEIGHT*SIN(ANG)
      DY=DY+HEIGHT*COS(ANG)
      CALL PLOT(DX,DY,2)

      DX=DX-WIDTH*COS(ANG)
      DY=DY-WIDTH*SIN(ANG)
      CALL PLOT(DX,DY,2)

      DX=DX+HEIGHT*SIN(ANG)
      DY=DY-HEIGHT*COS(ANG)
      CALL PLOT(DX,DY,2)

      IF(H.EQ.0.) RETURN
      IF(H.LT.0.) THEN
        DO 10 X=XO+WIDTH,XO,H
          CALL PLOT(X,YO,3)
          CALL PLOT(X,YO+HEIGHT,2)
10      CONTINUE
      ENDIF
      IF(H.GT.0.) THEN
        DO 20 Y=YO,YO+HEIGHT,H
          CALL PLOT(XO,Y,3)
          CALL PLOT(XO+WIDTH,Y,2)
20      CONTINUE
      ENDIF

      RETURN
      END
```

CONCLUSION

The program GRAPHER was designed as a tool to produce three different types of graphs, that are used most of the time for graphical evaluation of collected data. The graphs of normal distribution, linear regression, and simple connection of points on the graph, providing multiple lines on the same graph, are presented. The user is provided with a simple guide that will help him to start using this program. Additional information is also provided in multiple choices on screen of the terminal.

Examples of the types of the graphs are given, program listing in VAX FORTRAN 77 is also enclosed.